TITLE: METHODS, SYSTEMS, AND DATA STRUCTURES FOR PERFORMING SEARCHES ON THREE DIMENSIONAL OBJECTS
INVENTORS' NAMES: Karthik Ramani et al.
ATTORNEY DOCKET NO.: 1165.021US1
CLIENT REFERENCE NO.: P-02132.P1.US

100

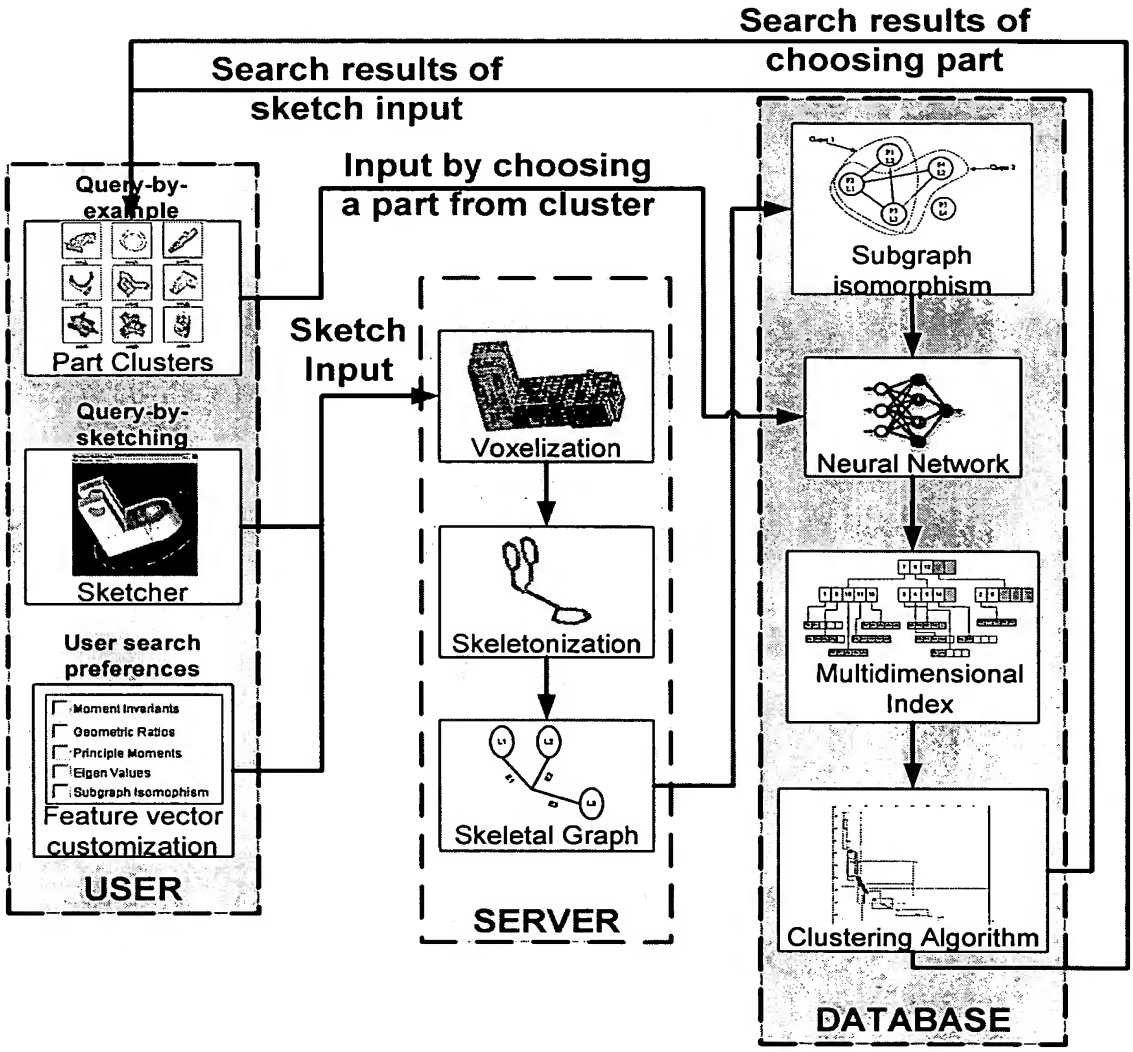


FIG. 1

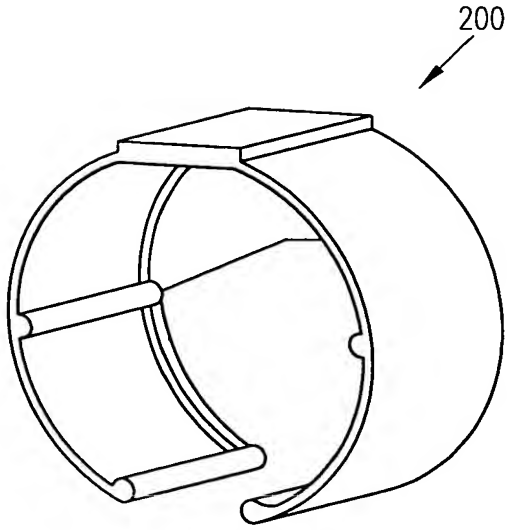


FIG. 2A

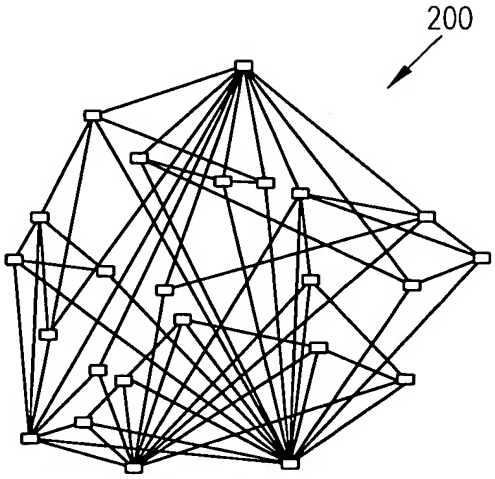


FIG. 2B

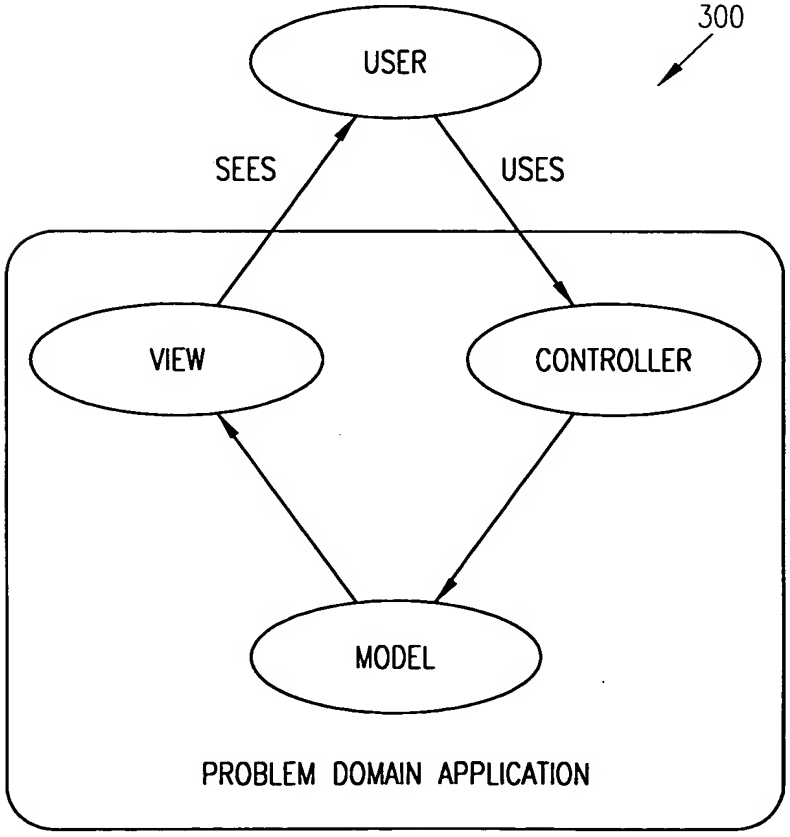


FIG. 3

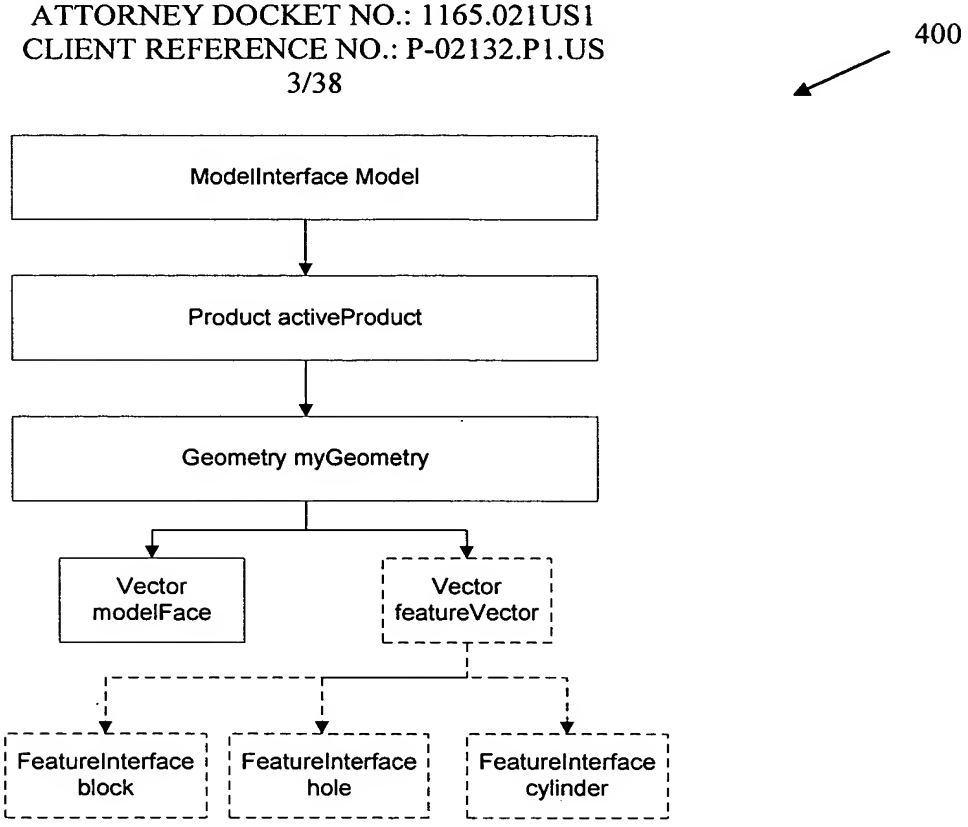

400
ModelInterface Model
Product activeProduct
Geometry myGeometry
Vector modelFace
Vector featureVector
FeatureInterface block
FeatureInterface hole
FeatureInterface cylinder

FIG. 4

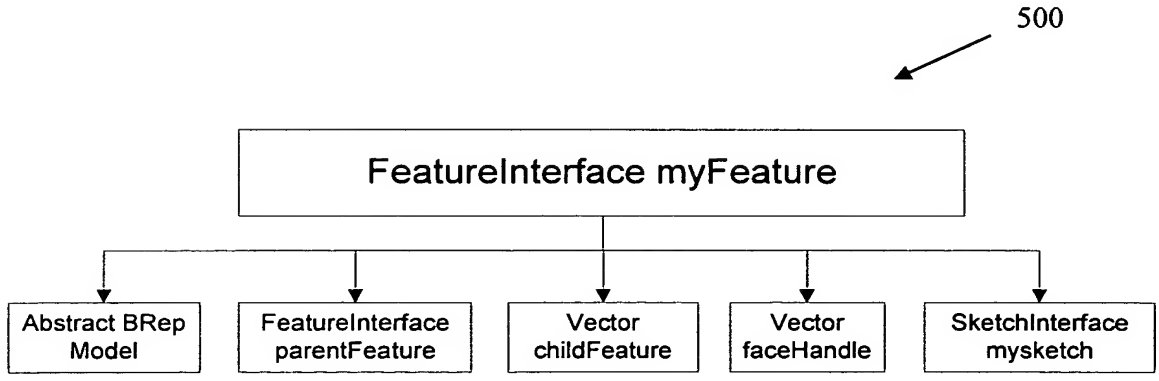

500
FeatureInterface myFeature
Abstract BRep Model
FeatureInterface parentFeature
Vector childFeature
Vector faceHandle
SketchInterface mysketch

FIG. 5

600

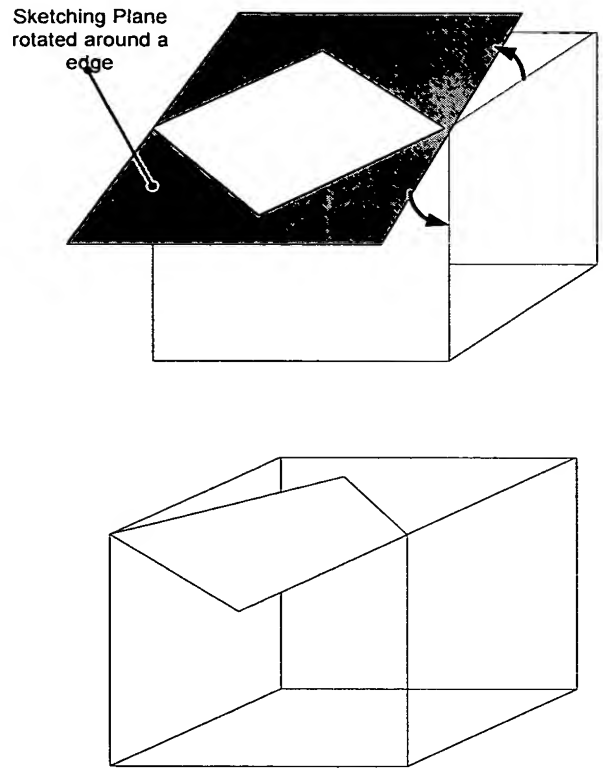


FIG. 6

700

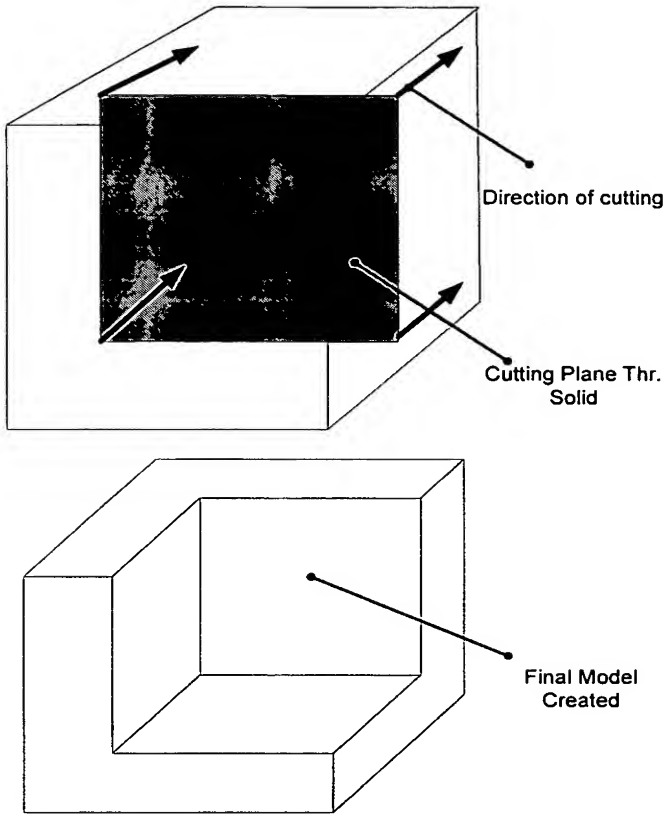

Direction of cutting
Cutting Plane Thr. Solid
Final Model Created

FIG. 7

800

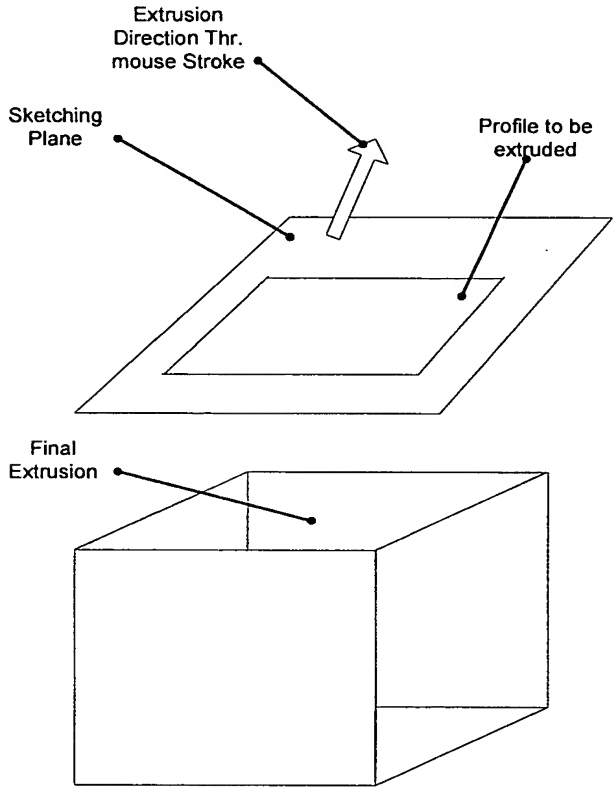

Extrusion Direction Thr. mouse Stroke
Sketching Plane
Profile to be extruded
Final Extrusion

FIG. 8

900

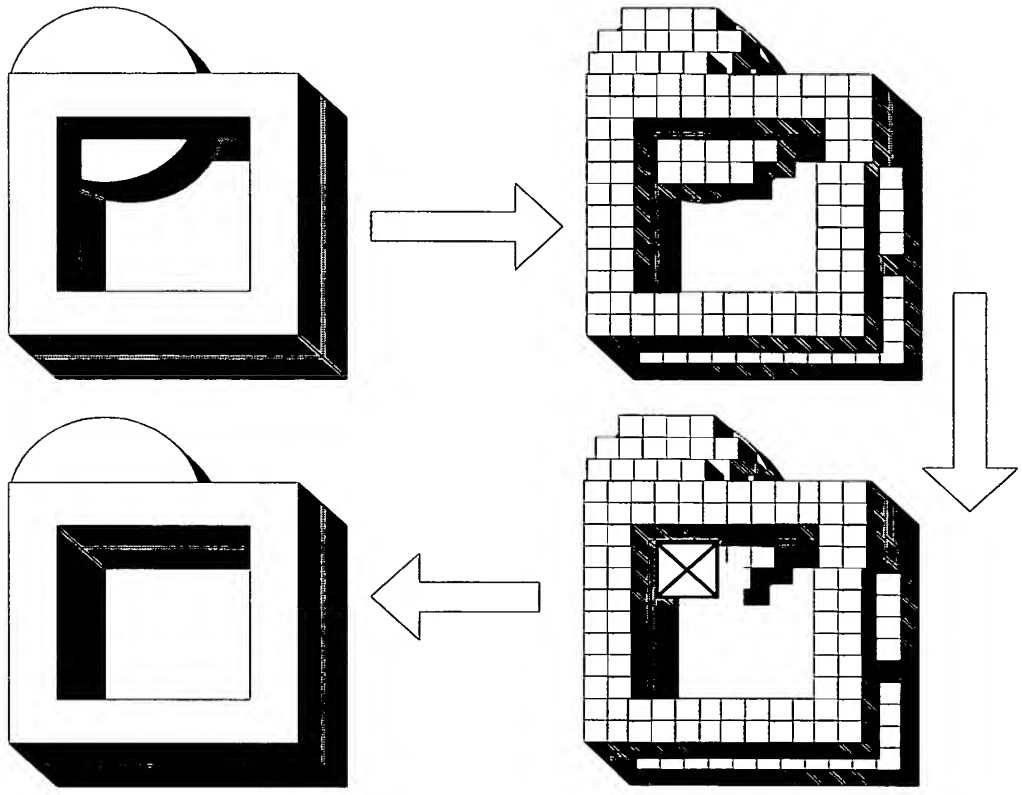
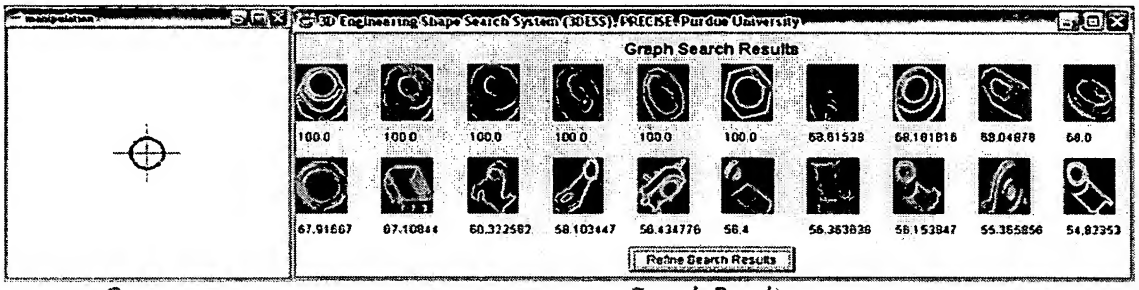


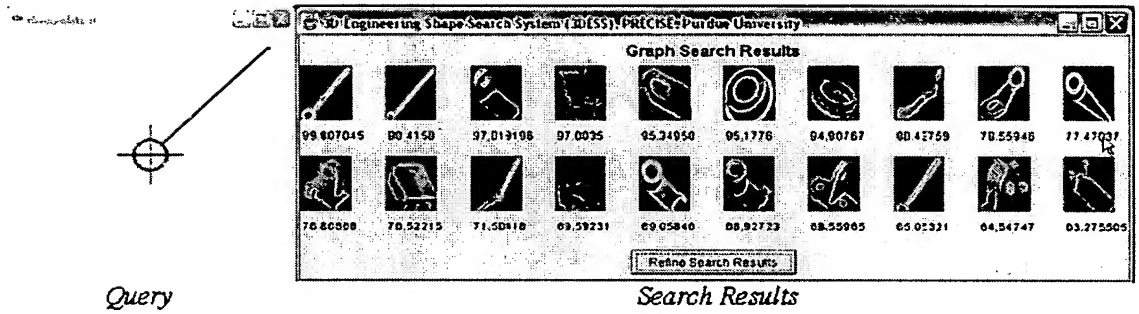
FIG. 9

TITLE: METHODS, SYSTEMS, AND DATA STRUCTURES FOR PERFORMING SEARCHES ON THREE DIMENSIONAL OBJECTS
INVENTORS' NAMES: Karthik Ramani et al.
ATTORNEY DOCKET NO.: 1165.021US1
CLIENT REFERENCE NO.: P-02132.P1.US

1000



(i) Initial Skeleton Query



(ii) Modified Skeleton Query

FIG. 10

1100

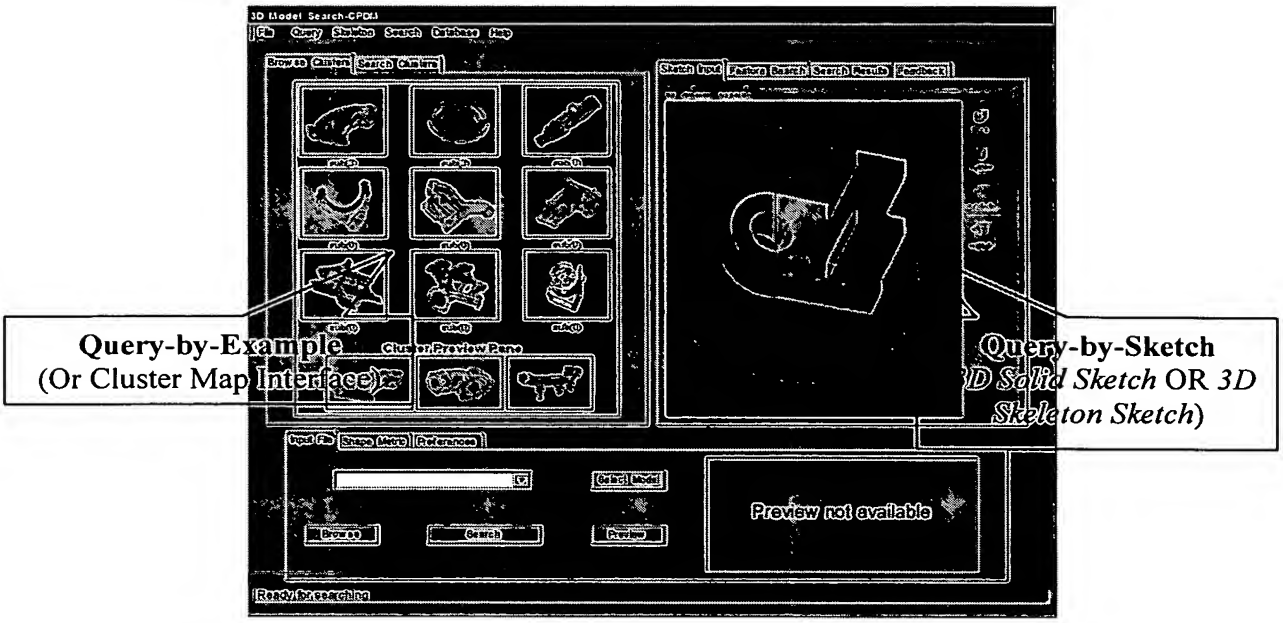


FIG. 11

1200

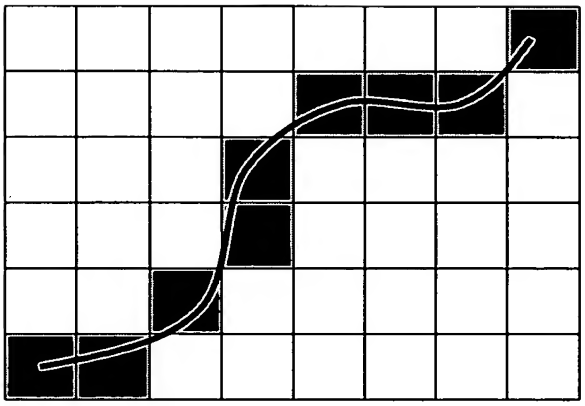


FIG. 12

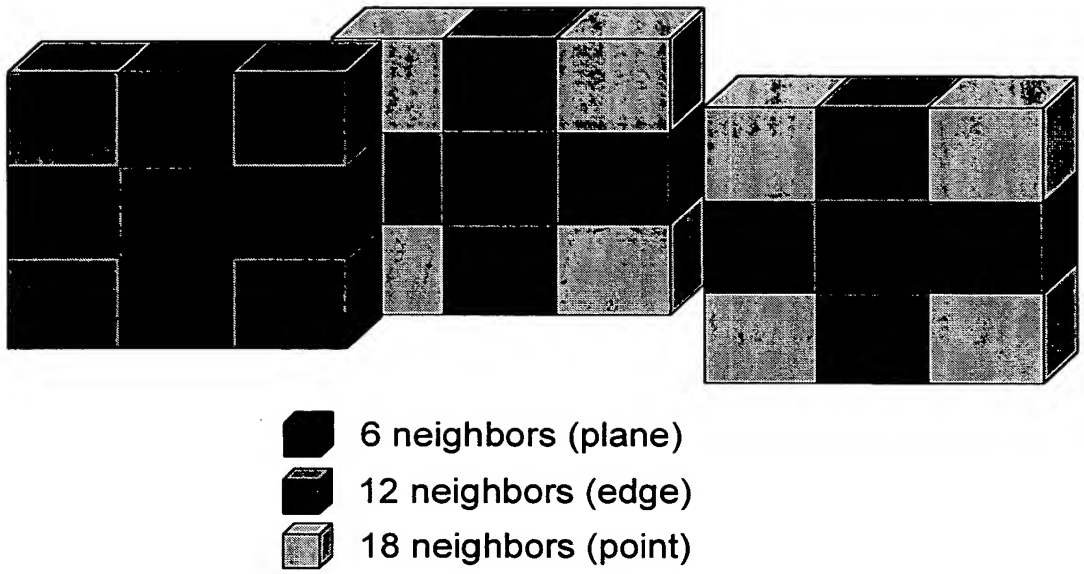

1300
6 neighbors (plane)
12 neighbors (edge)
18 neighbors (point)

FIG. 13

1400

```
Read in B-Rep model M
Assign FaceIDs F to all faces in M
Assign EdgeIDs E to all faces in M
Set voxel size S
Find B-Rep model bounding box
Find discrete bounding box coordinates X, Y ,Z
Create voxel text file T

for i in X in increment of S:
  for j in Y in increment of S:
    for k in Z in increment of S:
                construct voxel v of size S using ACIS
                test intersection of v with M using ACIS
                if v intersects with M:
                        store "1" in T
                        find faces intersecting with v using ACIS
                        find edges intersecting with v using ACIS
                        store F and E for all intersecting faces in v
                else:
                        store "0" in T
```

FIG. 14

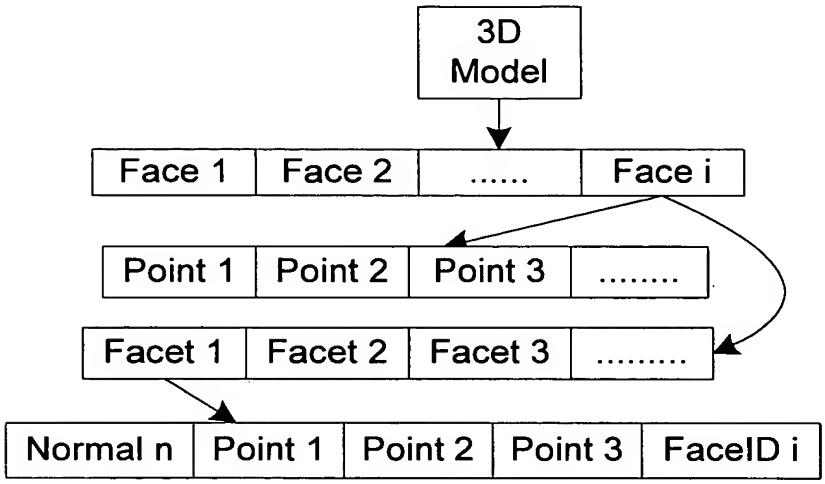

1500
3D Model
Face 1
Face 2
.....
Face i
Point 1
Point 2
Point 3
.....
Facet 1
Facet 2
Facet 3
.....
Normal n
Point 1
Point 2
Point 3
FaceID i

FIG. 15

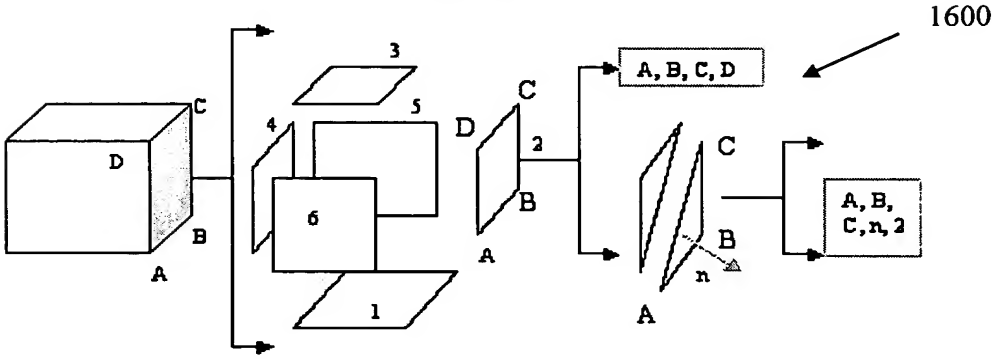

1600
A, B, C, D
A, B, C, n, 2
C
D
B
A
1
2
3
4
5
6
n

FIG. 16

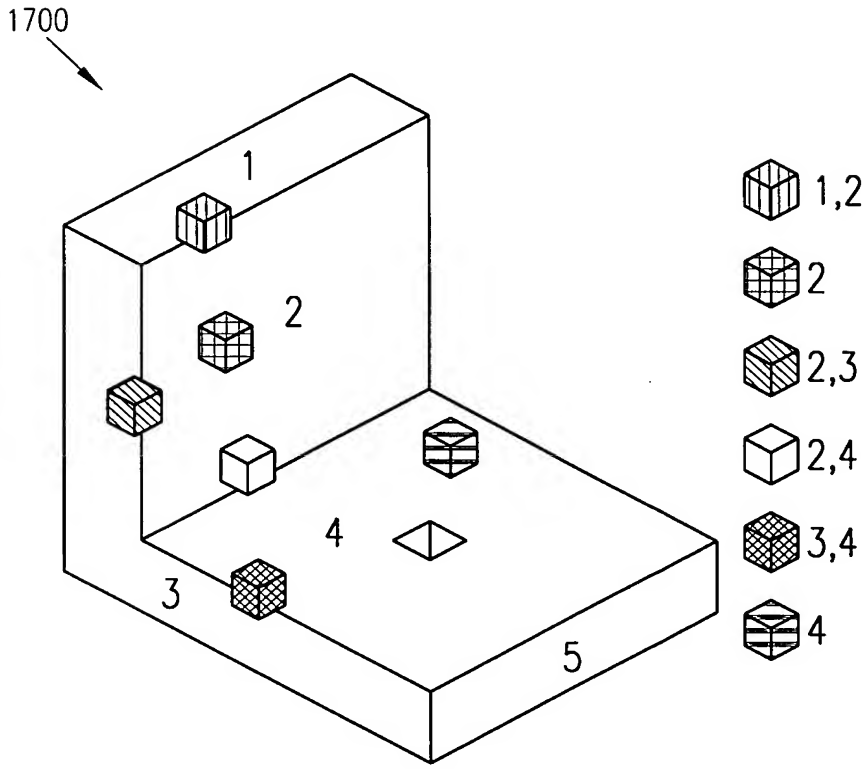


FIG. 17

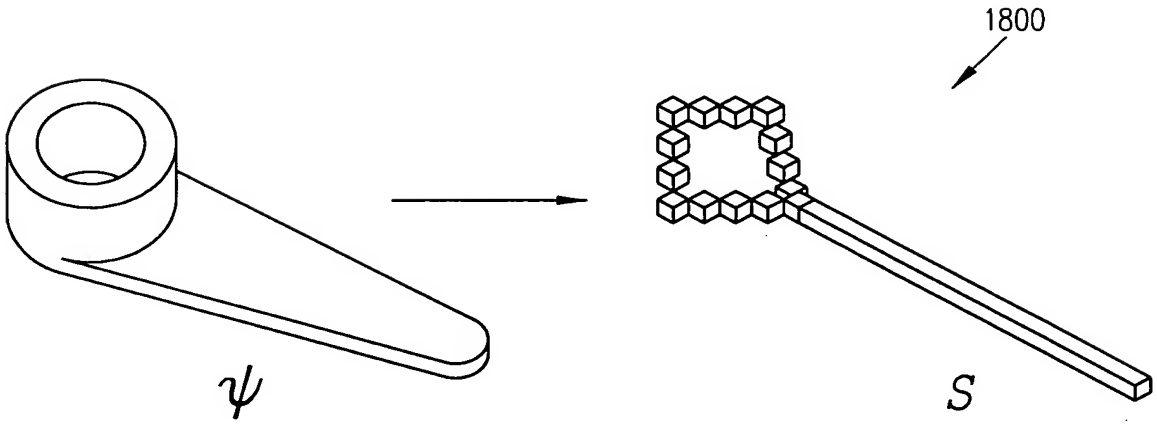


FIG. 18

1900

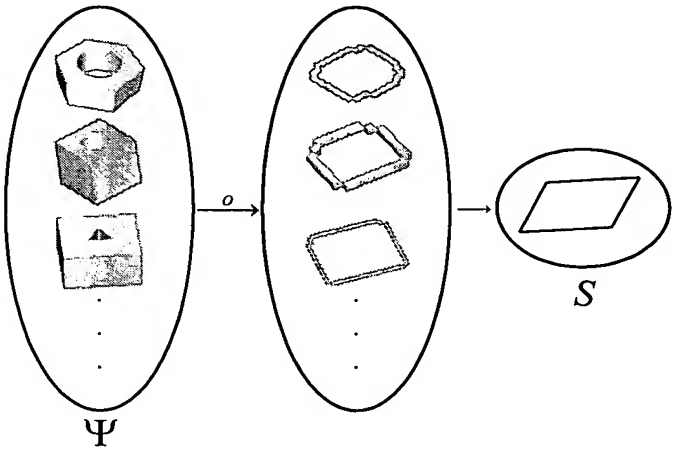


FIG. 19

2000

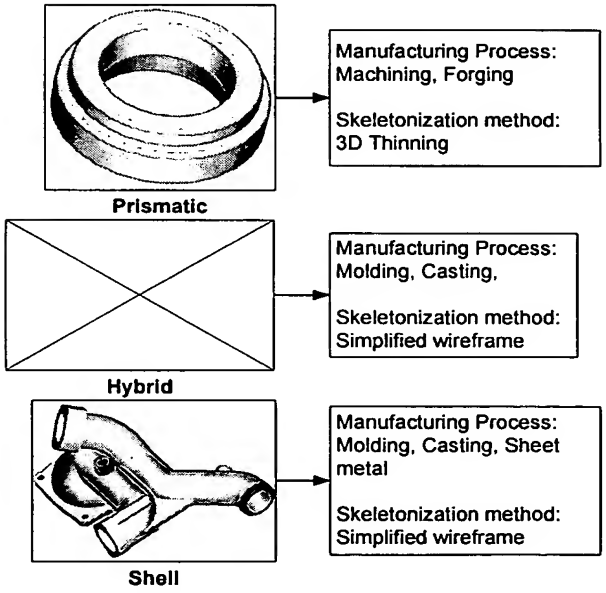


FIG. 20

2100

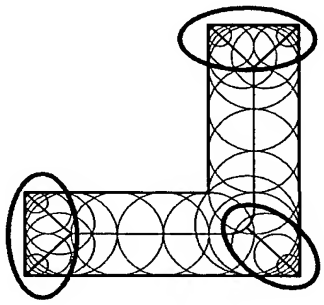


FIG. 21

2200

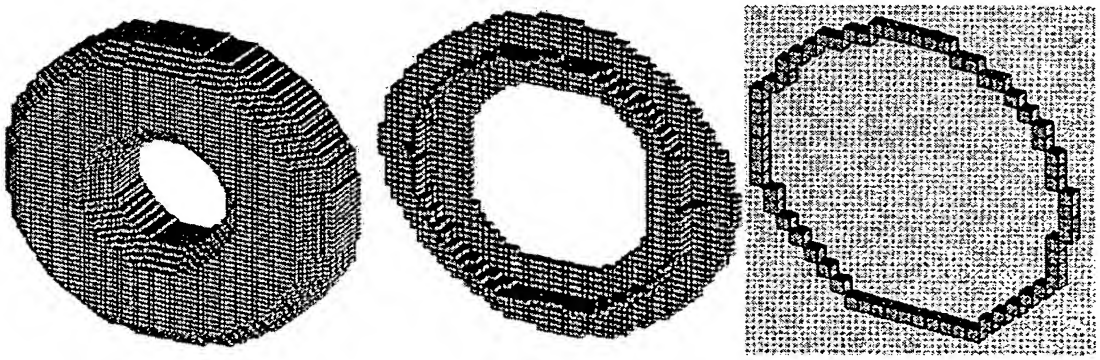


FIG. 22

2300

FIG. 23

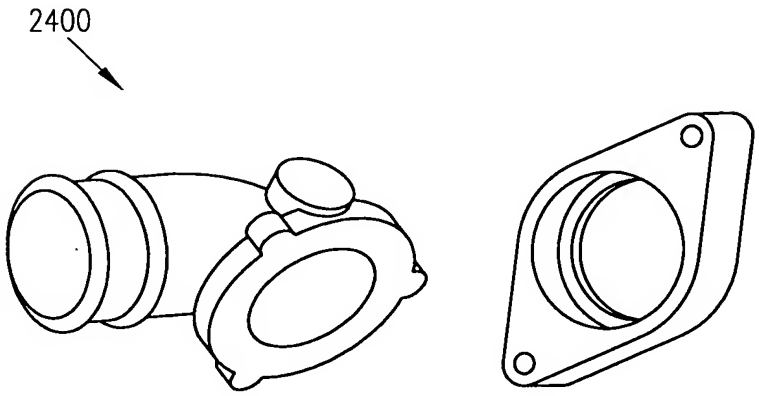

2400

FIG. 24

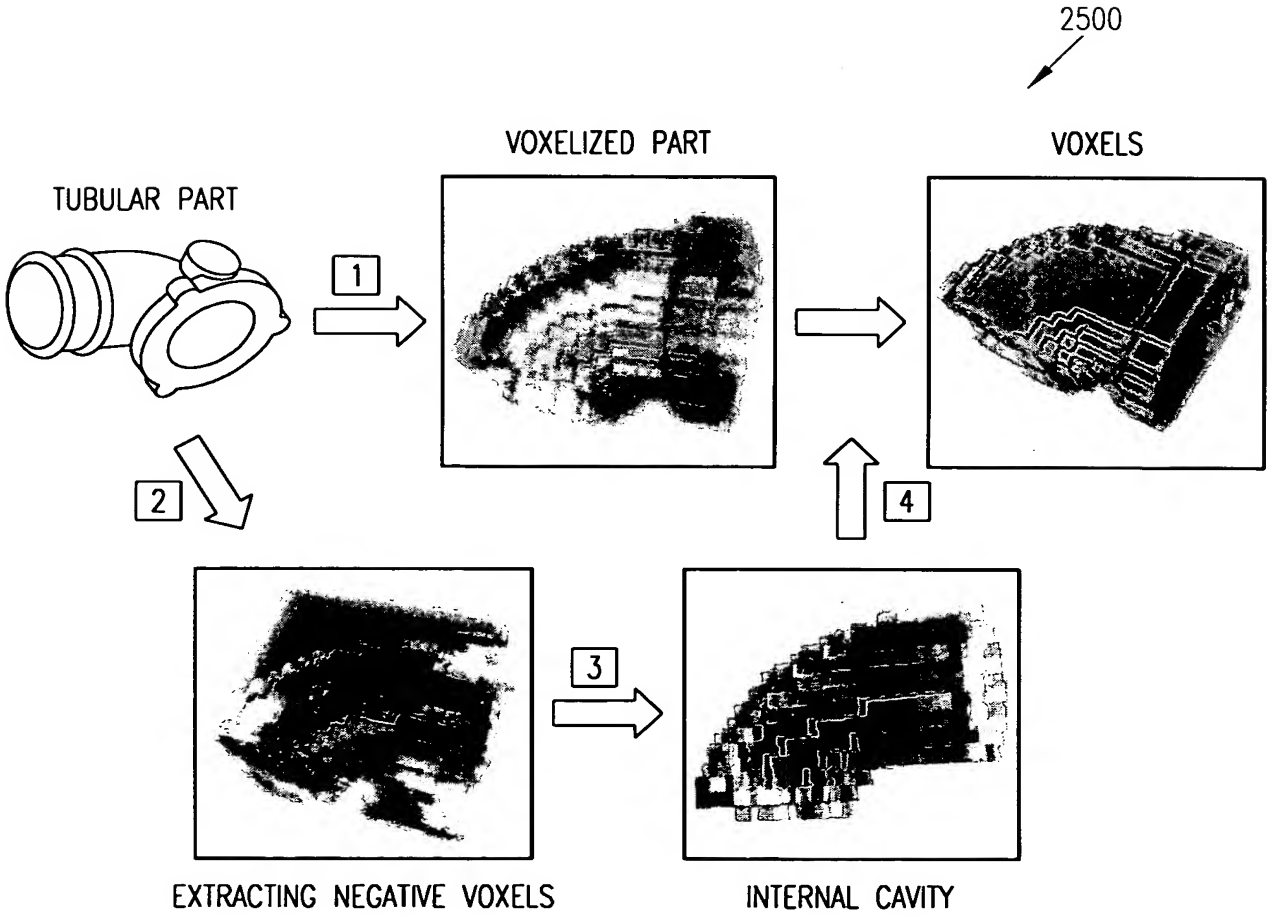

2500
TUBULAR PART
VOXELIZED PART
VOXELS
1
2
3
4
EXTRACTING NEGATIVE VOXELS
INTERNAL CAVITY

FIG. 25

2600

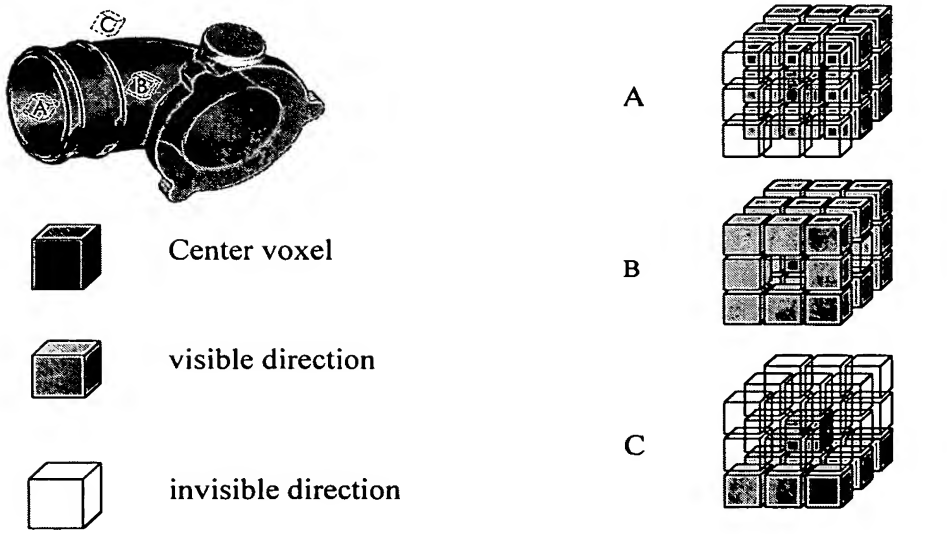


FIG. 26

2700

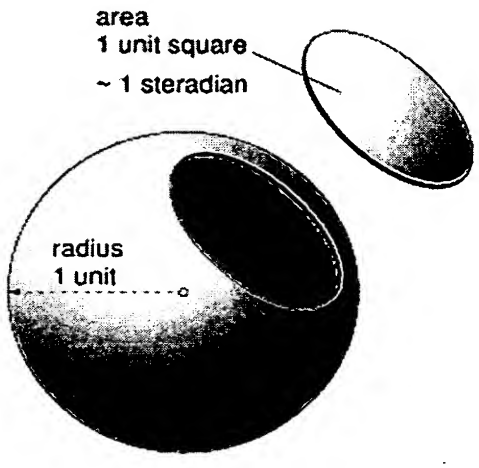


FIG. 27

2800

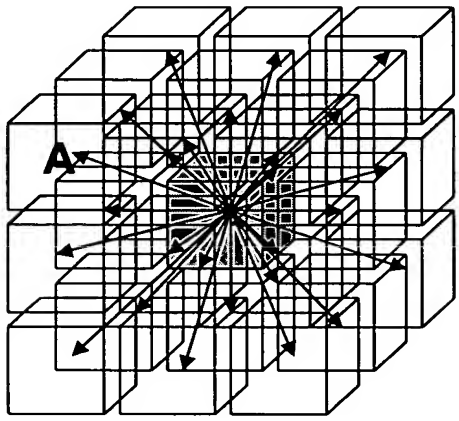


FIG. 28

2900

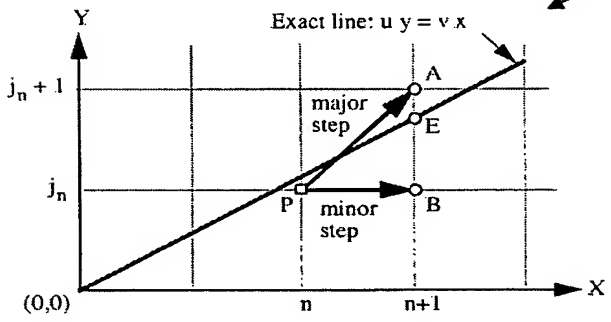


FIG. 29

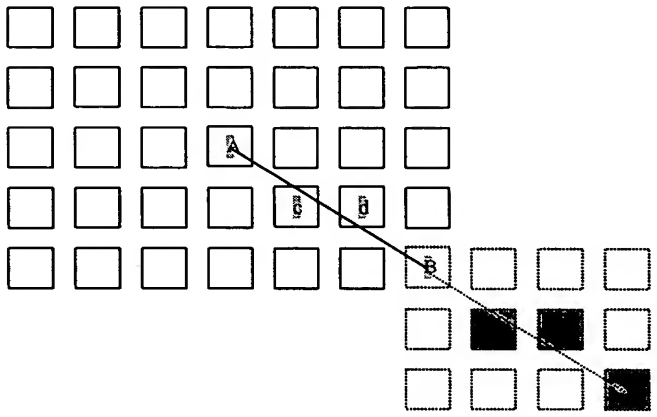


FIG. 30

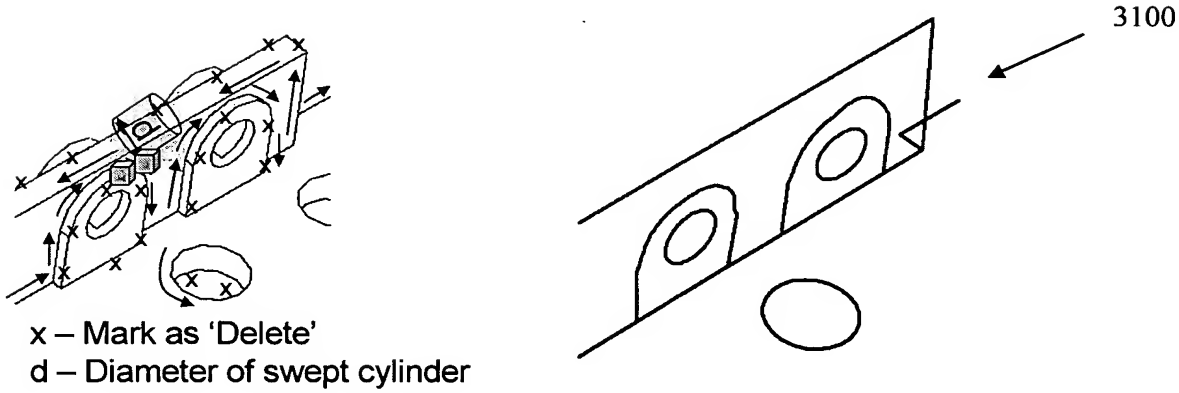


FIG. 31

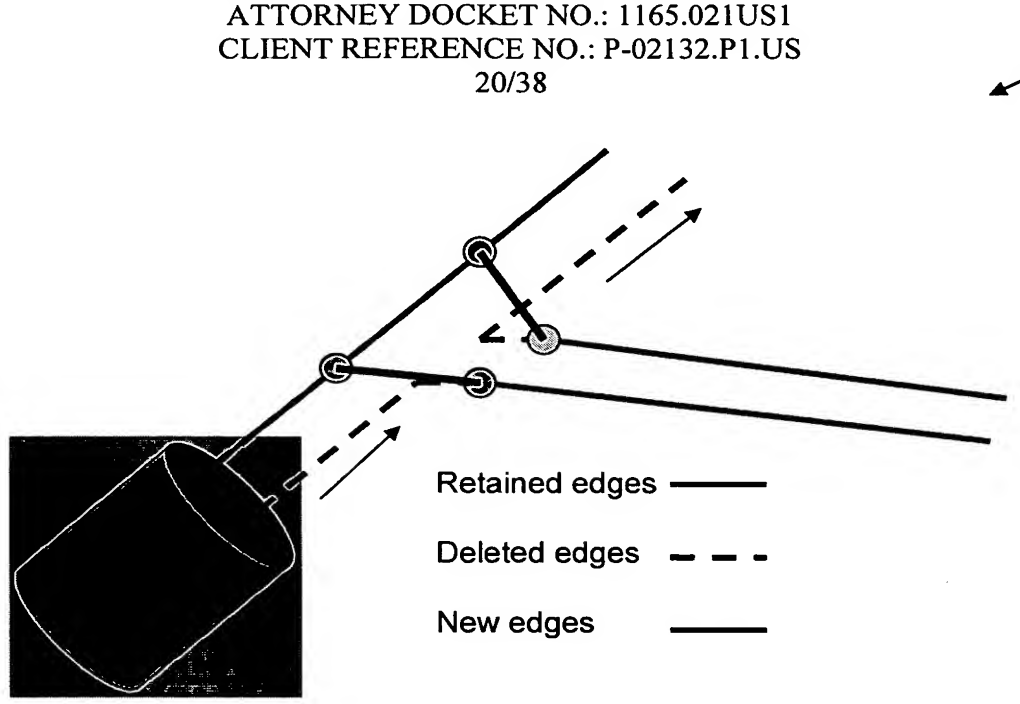


FIG. 32

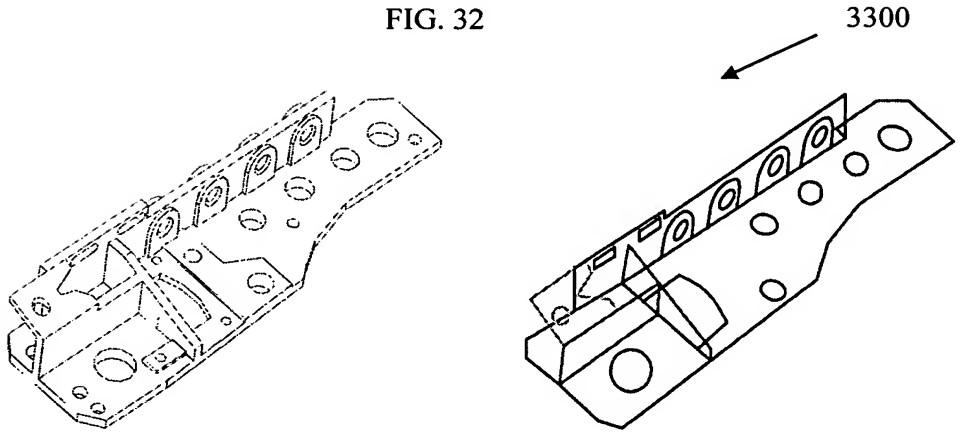


FIG. 33

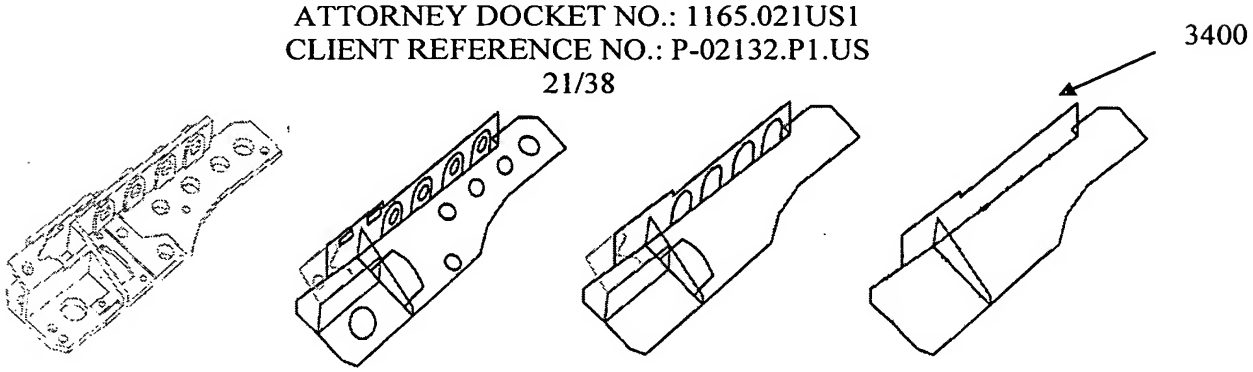


FIG. 34

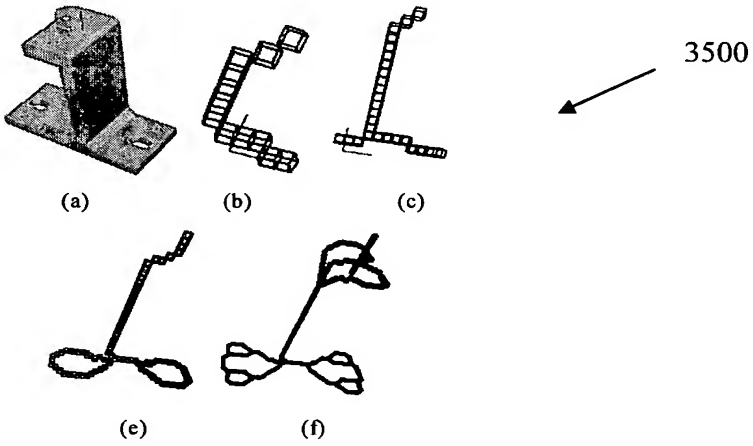


FIG. 35: (a) 3D model, (b) Levels of detail (LOD) for n = 2, (c) LOD for n = 3, (d) LOD for n = 4, (d) LOD for n = 5

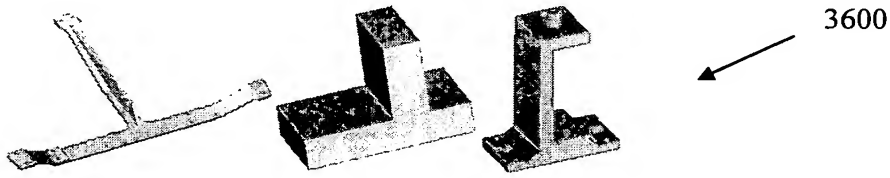


FIG. 36

3700

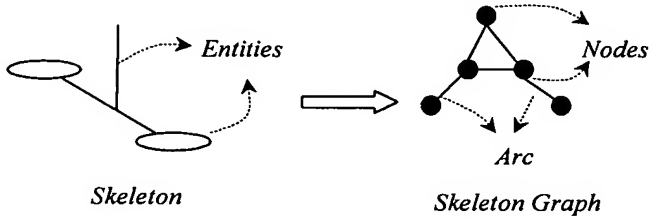


FIG. 37

3800

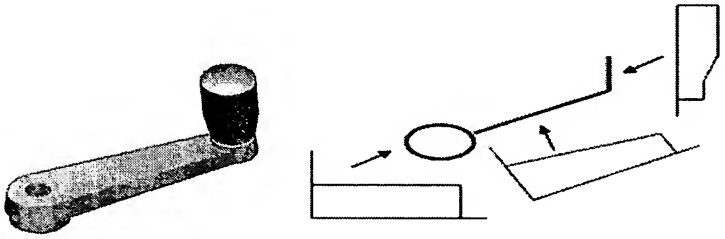


FIG. 38

3900

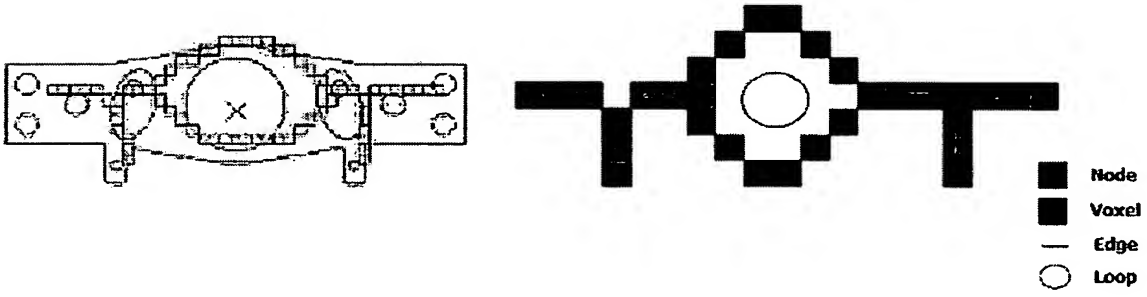


FIG. 39

4000

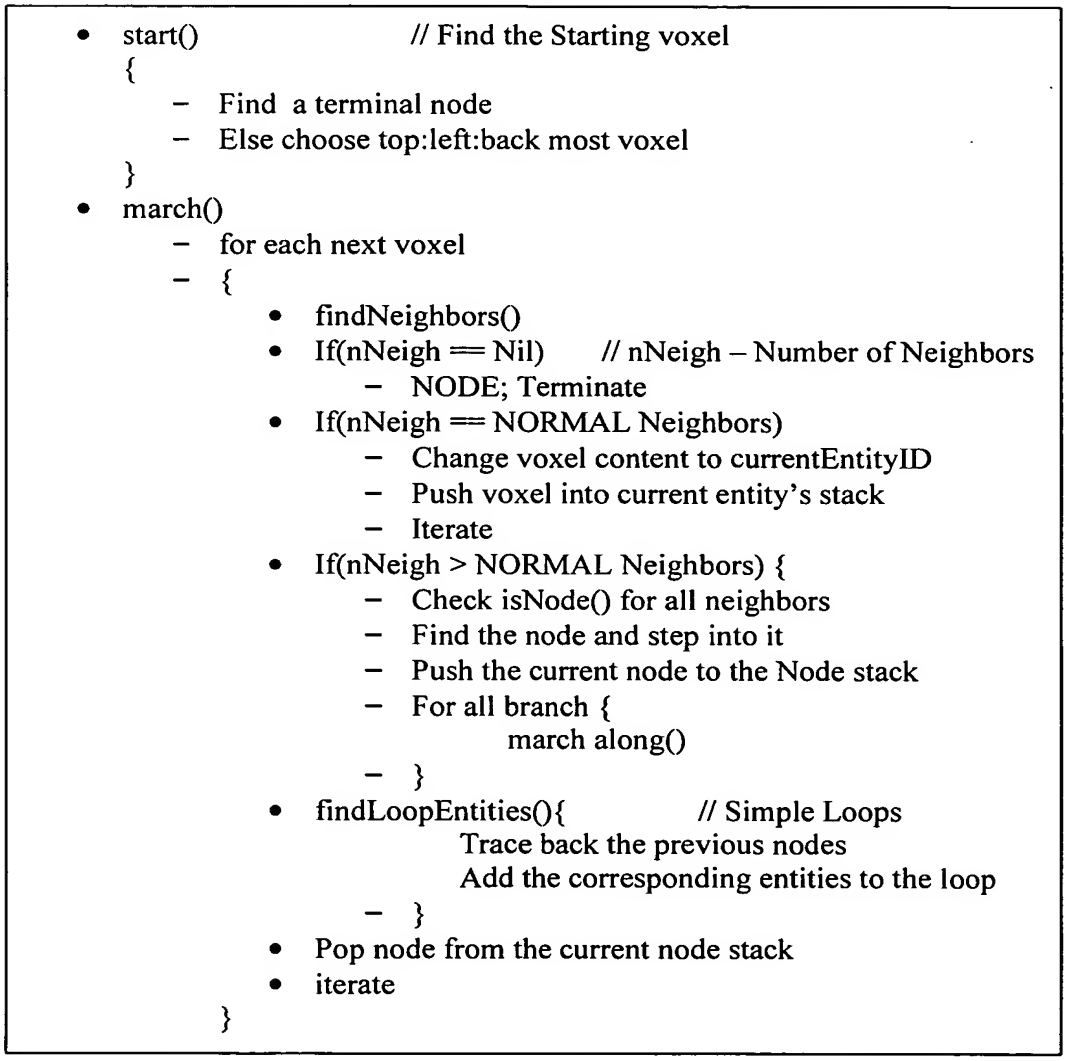

• start() // Find the Starting voxel
{
– Find a terminal node
– Else choose top:left:back most voxel
}
• march()
– for each next voxel
– {
• findNeighbors()
• If(nNeigh == Nil) // nNeigh – Number of Neighbors
– NODE; Terminate
• If(nNeigh == NORMAL Neighbors)
– Change voxel content to currentEntityID
– Push voxel into current entity's stack
– Iterate
• If(nNeigh > NORMAL Neighbors) {
– Check isNode() for all neighbors
– Find the node and step into it
– Push the current node to the Node stack
– For all branch {
march along()
– }
• findLoopEntities(){ // Simple Loops
Trace back the previous nodes
Add the corresponding entities to the loop
– }
• Pop node from the current node stack
• iterate
}

FIG. 40

4100

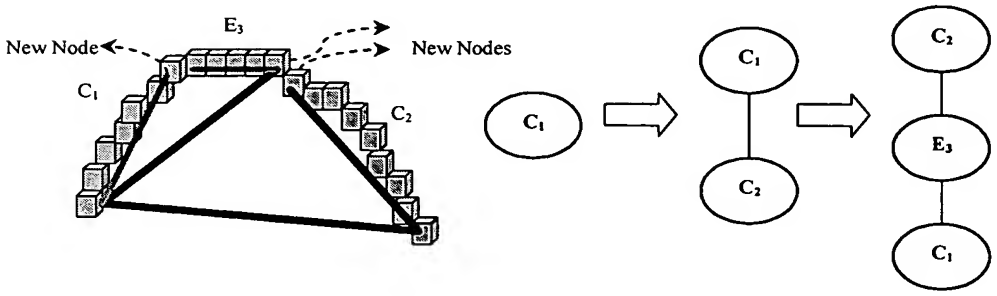


FIG. 41

4200

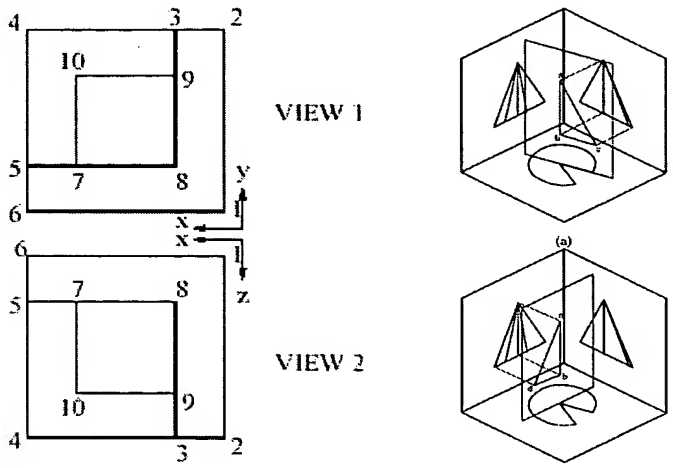


FIG. 42

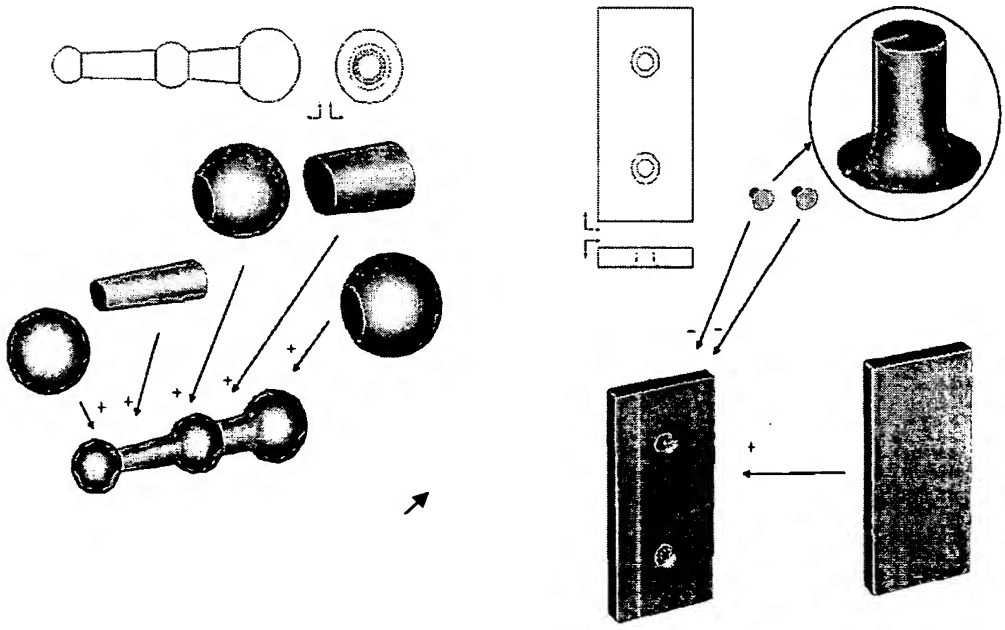


FIG. 43

4400

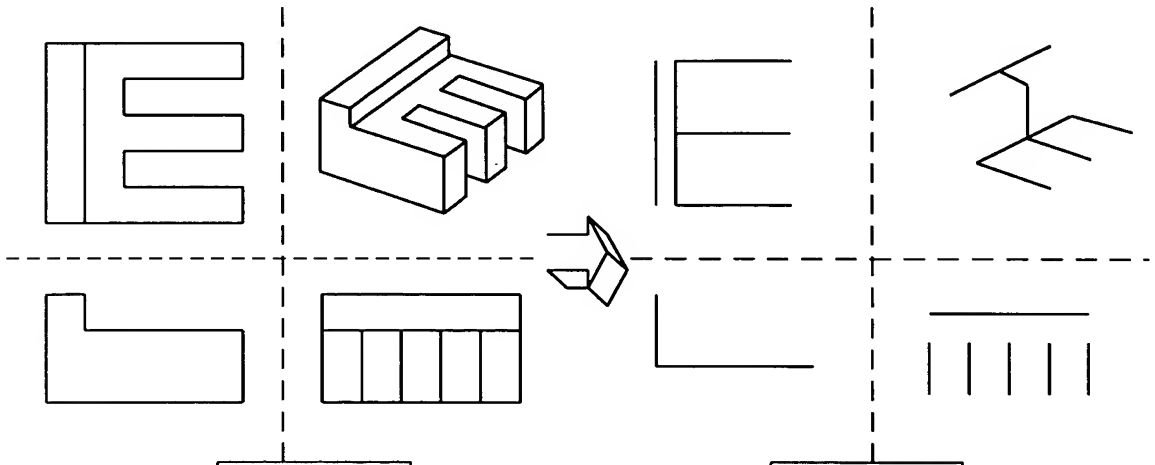


FIG. 44B

FIG. 44C

Fig. 44A

FIG. 44

4500

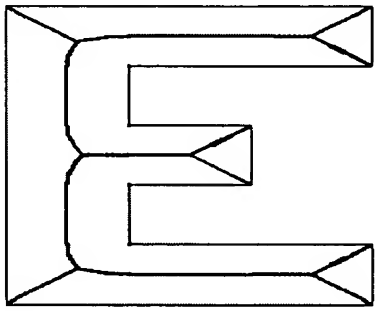


FIG. 45

4600

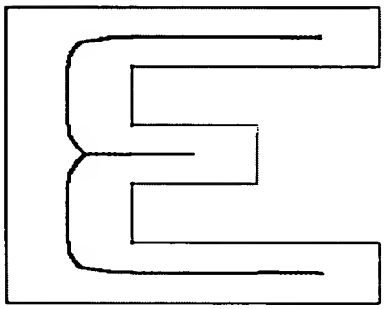


FIG. 46

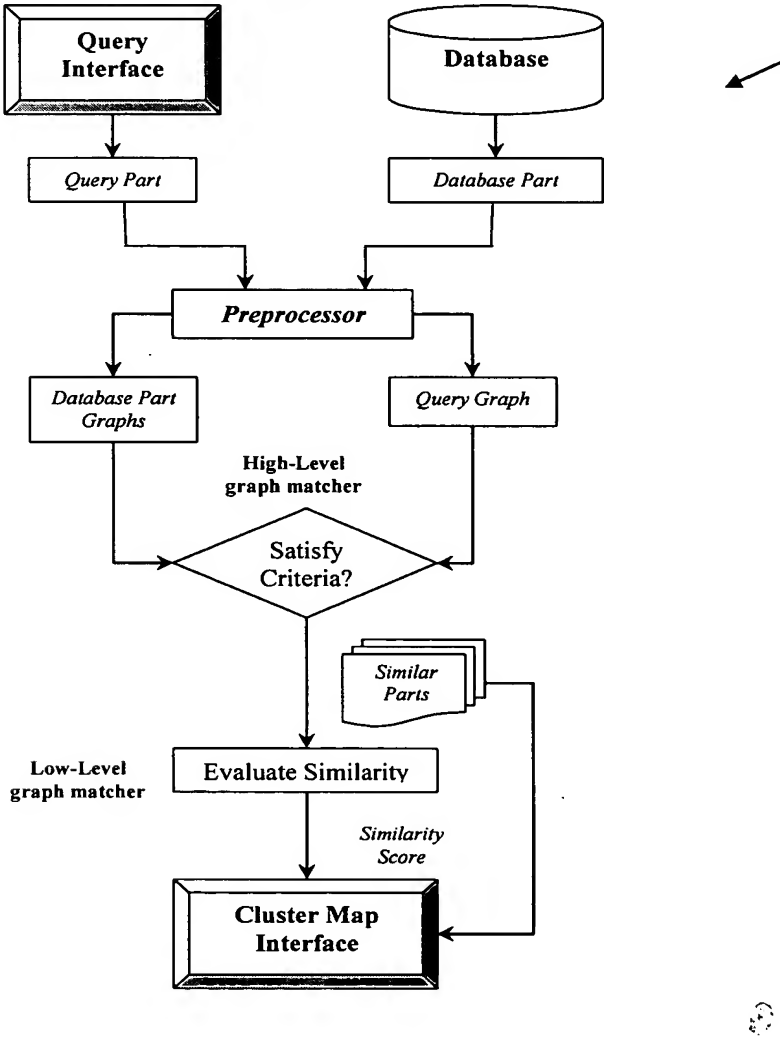

4700
Query Interface
Database
Query Part
Database Part
Preprocessor
Database Part Graphs
Query Graph
High-Level graph matcher
Satisfy Criteria?
Similar Parts
Low-Level graph matcher
Evaluate Similarity
Similarity Score
Cluster Map Interface

FIG. 47

4800

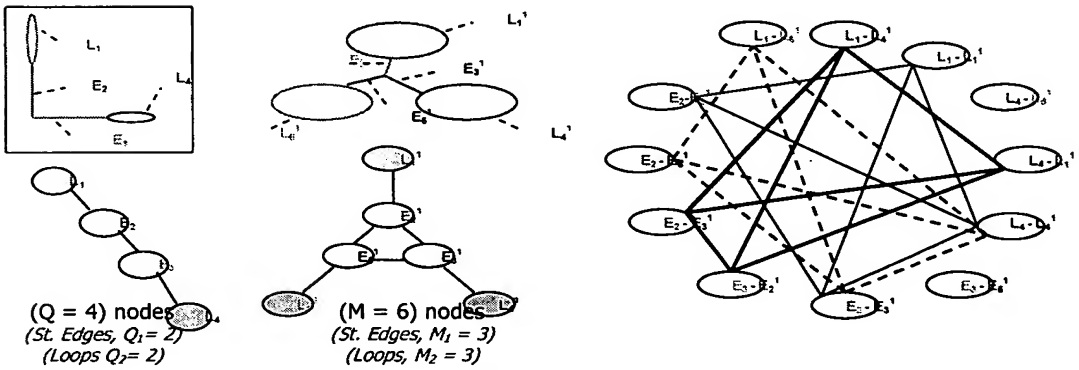


FIG. 48

4900

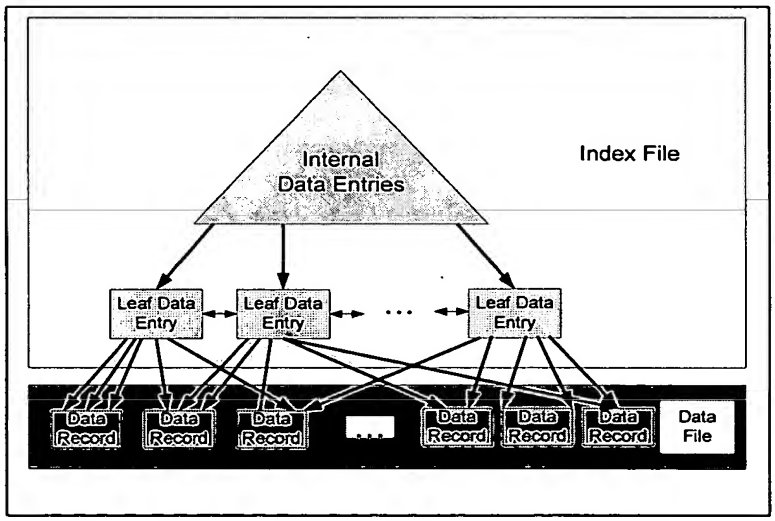


FIG. 49

5000

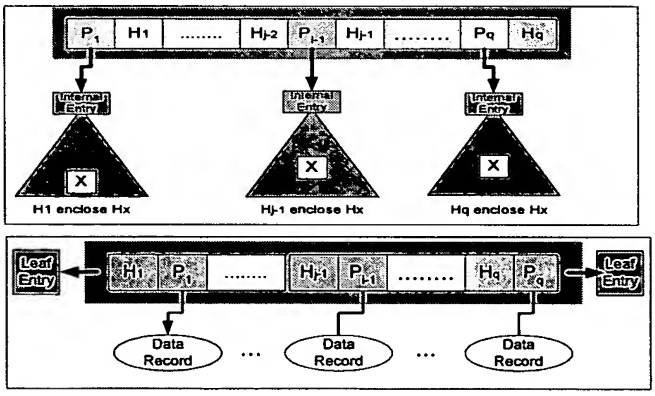


FIG. 50

5100

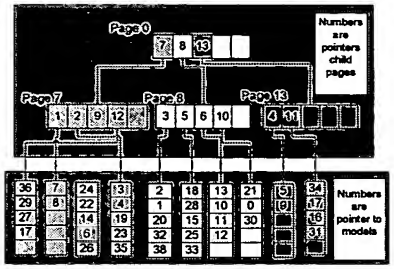


Figure 51A

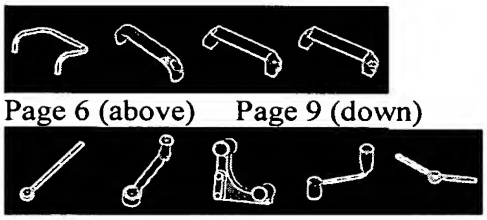


Figure 51B

FIGS. 51A and 51B

5200

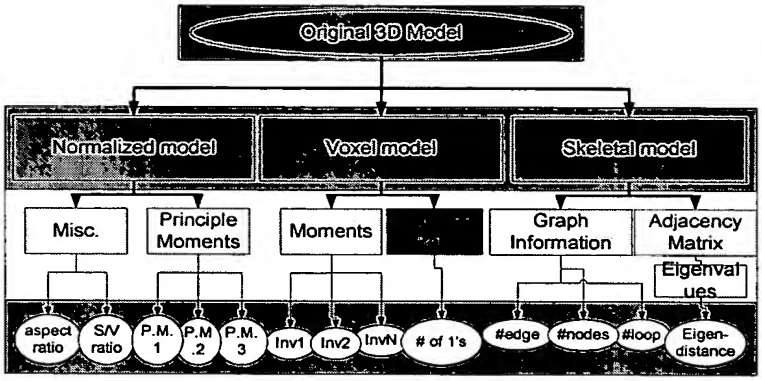


FIG. 52

5300

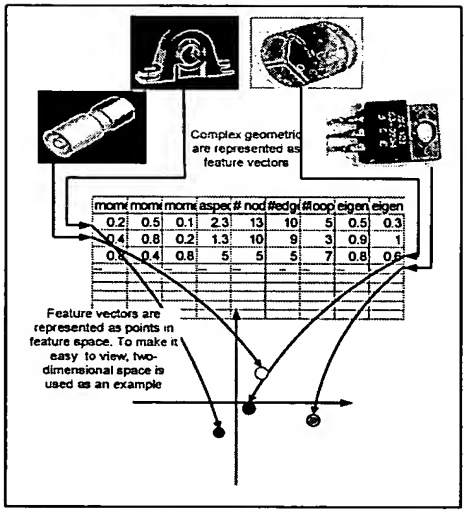


FIG. 53

5400

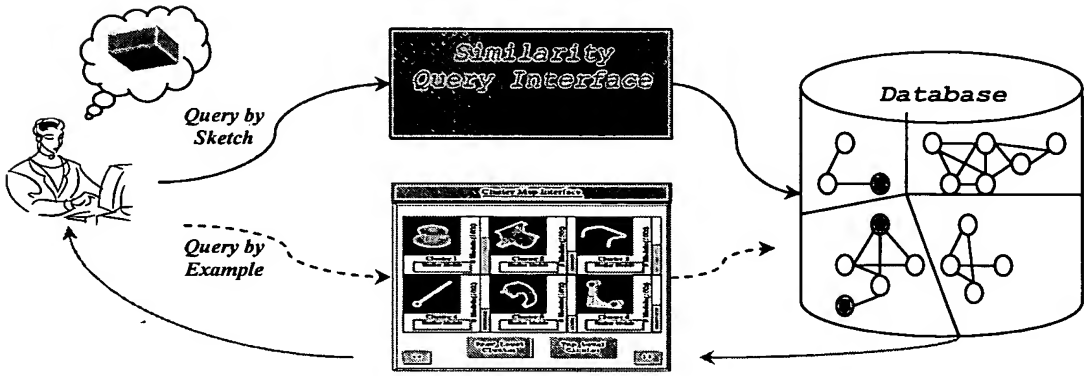


FIG. 54

5500

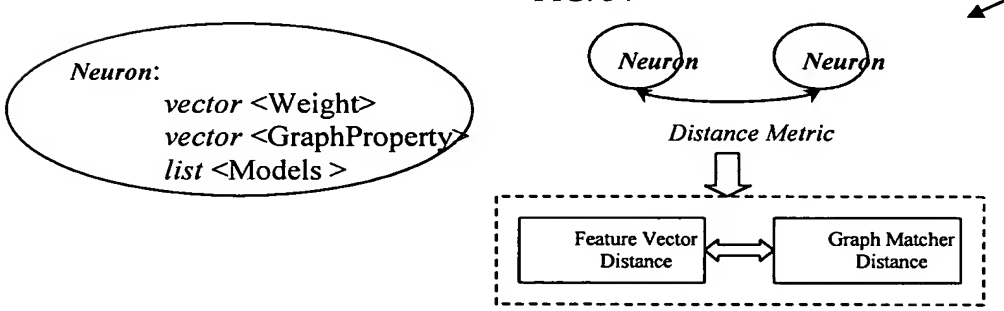


FIG. 55

5600

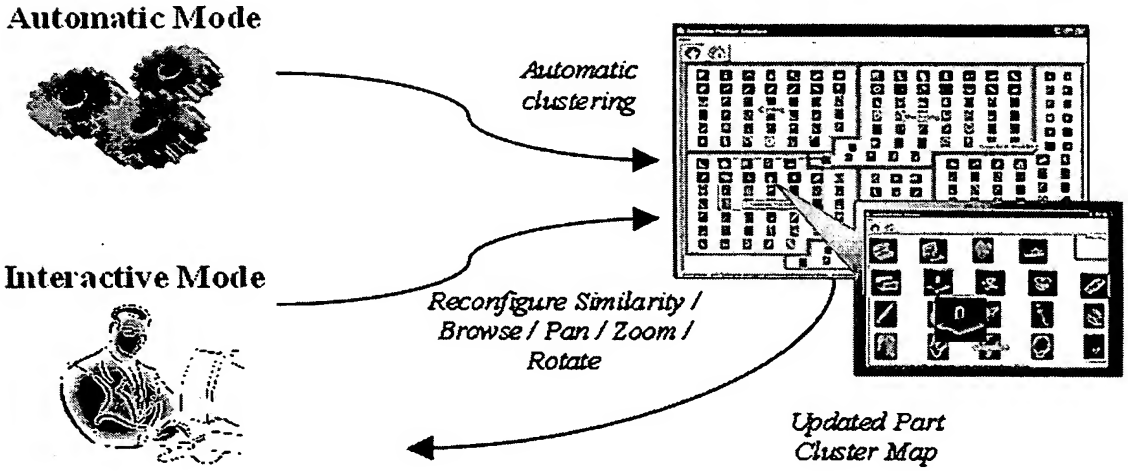


FIG. 56

5700

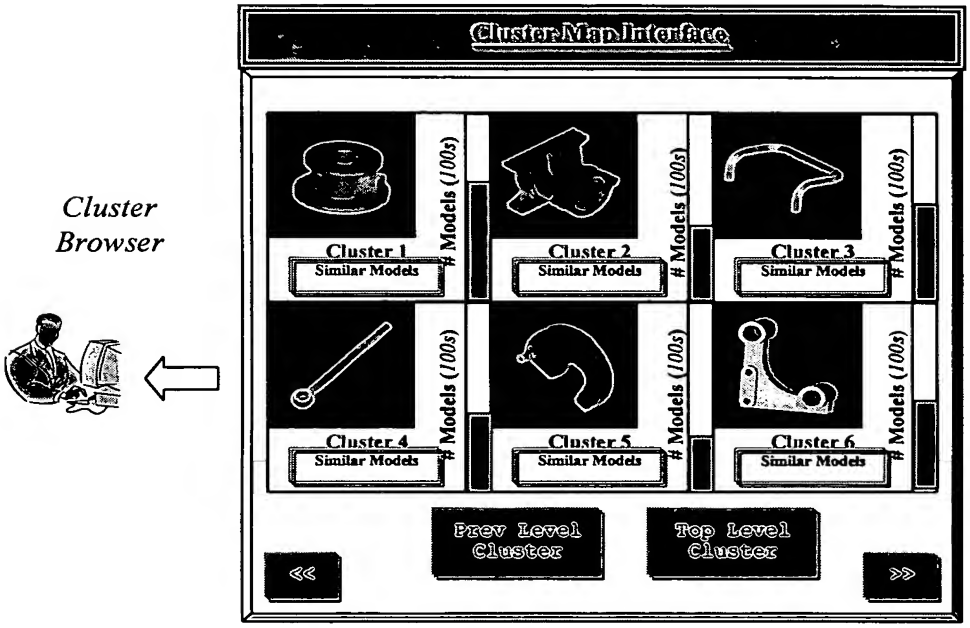


FIG. 57

5800

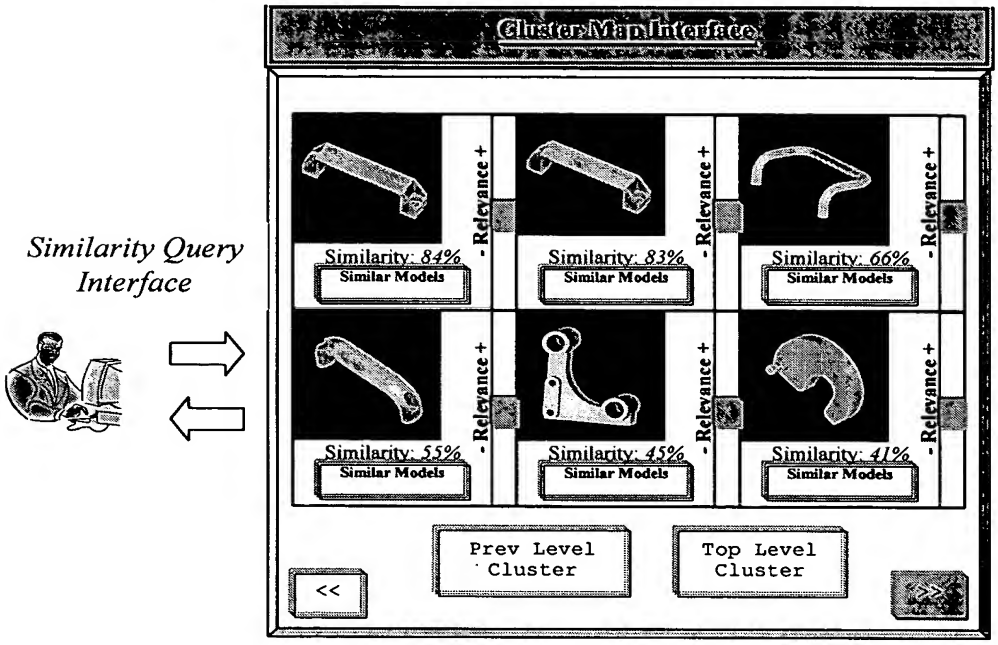

Similarity Query Interface
Cluster Map Interface
Similarity: 84%
Similarity: 83%
Similarity: 66%
Similarity: 55%
Similarity: 45%
Similarity: 41%
Similar Models
- Relevance +
Prev Level Cluster
Top Level Cluster
<<

FIG. 58

5900

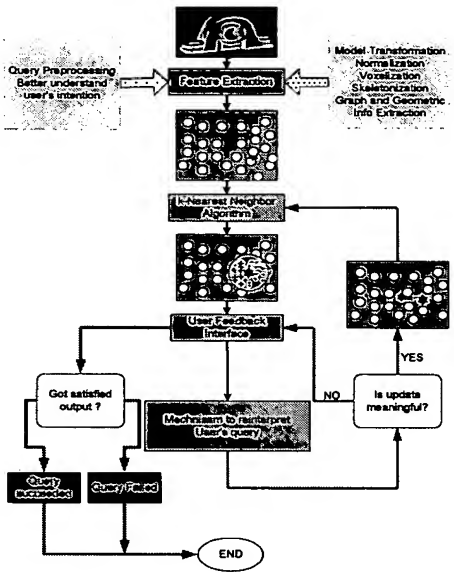


FIG. 59

6000

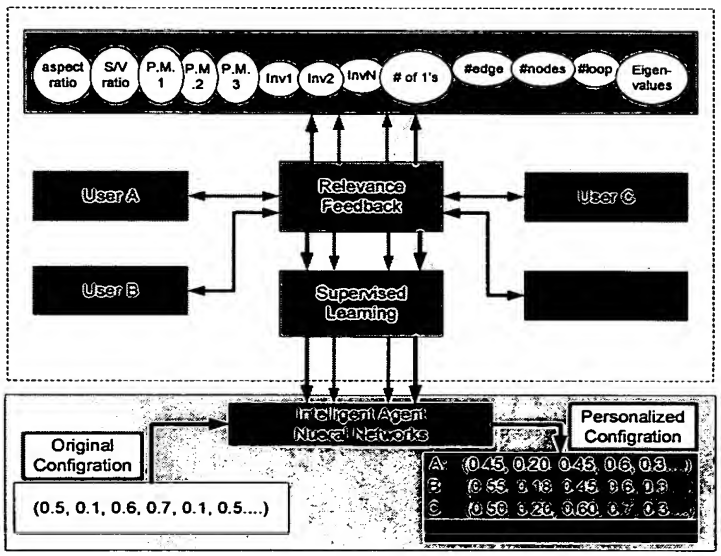


FIG. 60

6100

Database feature vector			Relevance Feedback	**Query Vector**		**Reconfiguration**
No.	F1	F2		F1	F2	
1	0.1	0.2		0.1	0.06	$W_1^{(1)} = W_1^{(0)}$
2	0.2	0.6		0.2	0.2	$W_2^{(1)} = 3 \times W_2^{(0)}$
3	0.3	0.4		0.3	0.13	
4	0.5	0.6		0.5	0.06	$W_1^{(1)} = W_1^{(0)}$
5	0.6	0.9		0.6	0.09	$W_2^{(1)} = 10 \times W_2^{(0)}$
6	0.7	1.0		0.7	0.10	
7	0.8	0.8				???
8	0.65	0.55				???

FIG. 61

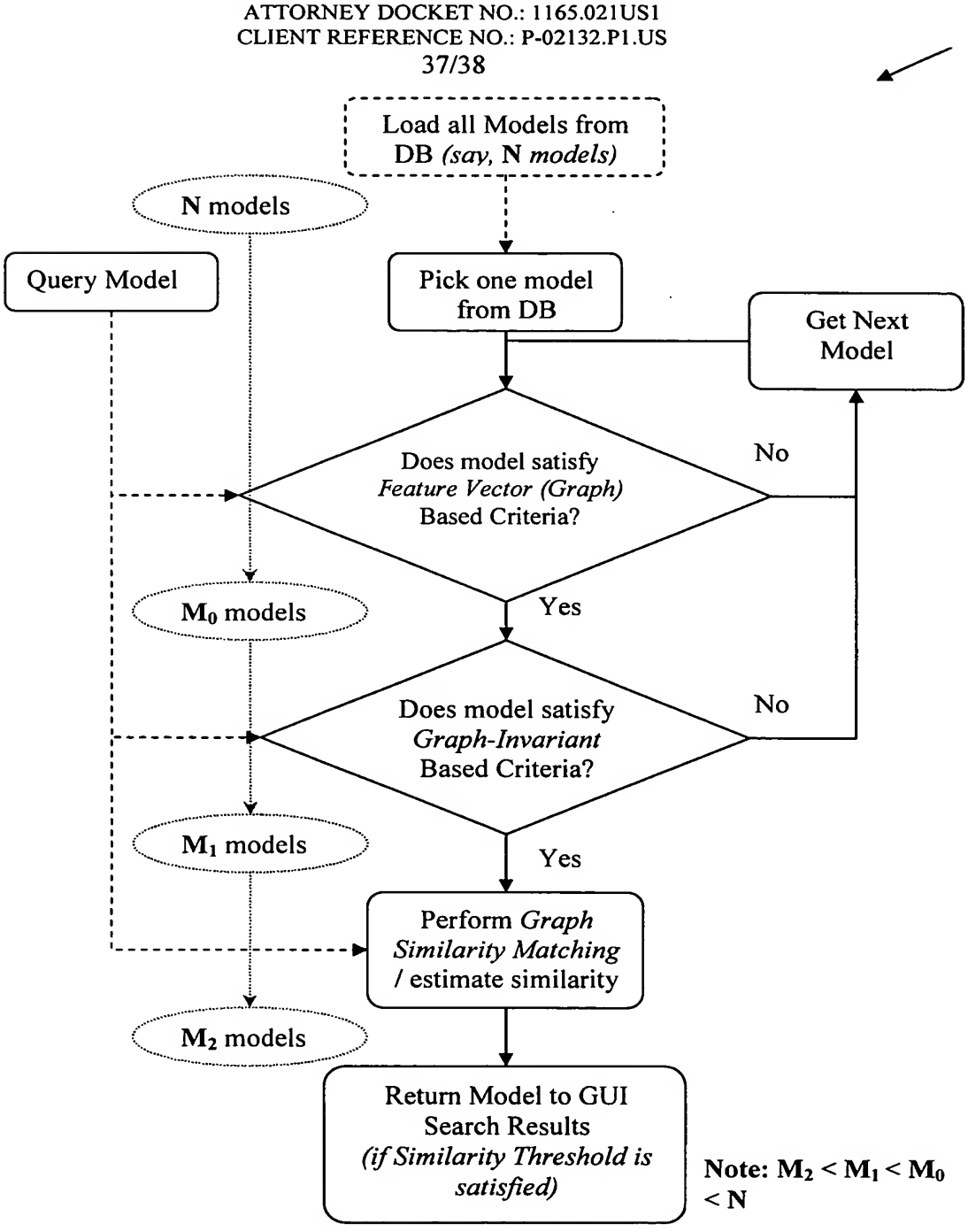
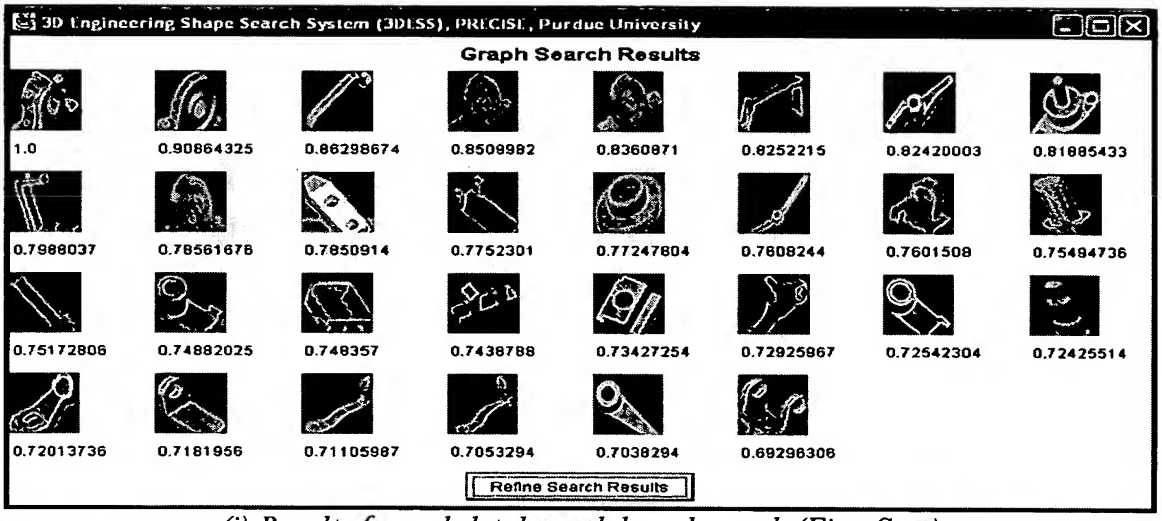

6200
Load all Models from DB (say, N models)
N models
Query Model
Pick one model from DB
Get Next Model
Does model satisfy Feature Vector (Graph) Based Criteria?
No
Yes
M0 models
Does model satisfy Graph-Invariant Based Criteria?
No
Yes
M1 models
Perform Graph Similarity Matching / estimate similarity
M2 models
Return Model to GUI Search Results (if Similarity Threshold is satisfied)
Note: M2 < M1 < M0 < N

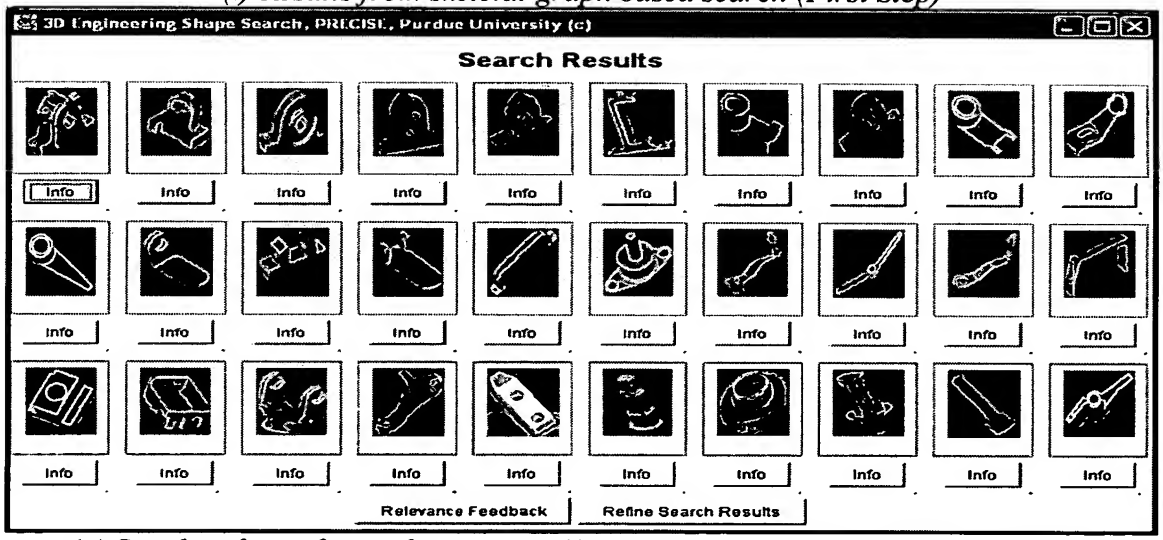
FIG. 62

TITLE: METHODS, SYSTEMS, AND DATA STRUCTURES FOR PERFORMING SEARCHES ON THREE DIMENSIONAL OBJECTS
INVENTORS' NAMES: Karthik Ramani et al.
ATTORNEY DOCKET NO.: 1165.021US1
CLIENT REFERENCE NO.: P-02132.P1.US

6300



(i) Results from skeletal graph based search (First Step)



(ii) Results after refining the query in (i) using moment invariants (Second Step)

FIG. 63